Window-based step-wise sequential phase partition for nonlinear batch process monitoring

Jingxiang Liu [a], Tao Liu [a, *], Jie Zhang [b]

a Institute of Advanced Control Technology, Dalian University of Technology, Dalian, 116024, P. R. China
b School of Chemical Engineering and Advanced Materials, Newcastle University, Newcastle upon Tyne NE1 7RU, UK

** Corresponding author. Tel: +86-411-84706465; Fax: +86-411-84706706*

Abstract: In this paper a window-based step-wise sequential phase partition method is proposed to improve monitoring performance for nonlinear batch processes with multiphase operations. The three-dimensional information matrix of batch operation is unfolded and normalized in the batch-wise direction to facilitate establishing the kernel principle component analysis (KPCA) models for phase partition. A moving window is introduced to improve the partition performance with respect to the process dynamics and time sequence for operation. Consequently, phase partition algorithms are developed for even- and uneven-length batch processes, respectively. Moreover, a traversal algorithm is given to determine the optimal choice of the KPCA parameters and the window size for phase partition. A numerical case and two industrial multiphase processes of injection molding and penicillin fermentation are used to demonstrate the effectiveness and merit of the proposed phase partition method.

Keywords: kernel principle component analysis (KPCA), multiphase batch process, uneven-length, sequential phase partition, multivariate statistical analysis

---

1 Introduction

For effective monitoring of industrial batch processes such as injection molding and pharmaceutical crystallization, multivariate statistical methods have been increasingly studied in the past decades, e.g. multi-way principal component analysis (MPCA) and multi-way partial least squares (MPLS) methods[1,2], obtaining evidently improved quality prediction and fault detection when compared to other approaches[3-6]. In the early developed MPCA or MPLS methods, one monitoring model was established for the whole batch process, regardless of the inner feature variations. In fact, the feature changes in many batch processes are related to different operation conditions. The idea of multiphase process monitoring was therefore proposed[7], which divides the process into several phases for modeling. It was elucidated that a set of monitoring models should be established to represent the multiphase nature of a batch process, in order to precisely identify different operations corresponding to individual phases[8,9].

In the recent years, a number of phase partition methods have been developed. Camacho et al presented a multiphase principal component analysis (MPPCA) method for batch process modeling and monitoring[10-12], that is, the partition results were tested at every possible points, determining a new phase if the modeling performance could be improved in terms of such a partition point. Clustering algorithms for multiphase partition were proposed in the references[13-18], where the K-means algorithm was commonly used in combination with different criteria. Lu et al proposed two stage-based sub-PCA and sub-PLS methods to perform ‘hard-partition’ on the loading matrices[13,14]. The stage-based method was further extended to be a ‘soft-partition’ type[15] and modified to solve the non-Gaussian problem relating to multiphase processes[16]. By comparison, it was proposed to use the angle defined between different score spaces to quantify the similarities between PCA models[18]. Note that in the developed clustering-based partition methods, the phase center was predefined and the time sequence of operation phases was not considered therein. In the sequel, the time segments belonging to different phases might be clustered into a single phase while certain points in one phase might be assigned to other phases based on the evaluation of process similarity. To solve the problem of time sequence involved with these clustering methods, a step-wise sequential phase partition (SSPP) algorithm was

proposed in the recent papers[19,20], where segments in the time sequence was automatically determined. For nonlinear process monitoring, kernel-based methods[21-24] were adopted in the recent years. Kernel-based clustering methods[25] have also been proposed whereas the drawbacks of clustering methods remained as yet. Singular points were taken as the key index to detect the phase changes in the existing references[26,27], demonstrating good effectiveness for multiphase processes with distinct features in different phases. Based on the cyclic repetition information of batch processes, a repetition factor was defined to partition the multiple phases of batch operation[28,29]. Note that the characteristics of process dynamics were not considered in most of the above phase partition methods, which, however, could be used to improve phase partition accuracy in practical applications.

Besides, it had been assumed in existing references that all batches had the same duration and the batch trajectories were synchronized for multiphase processes. In fact, the uneven-length problem is often involved with multiphase process operation in engineering applications, which brings difficulties for application of the developed process monitoring methods based on even-length batch analysis. A small number of process monitoring methods have been devoted to tackle the problem, which may be divided into two types: batch length synchronization based monitoring methods[30-32] and irregular phase partition based monitoring methods[33,34]. Besides, mixture model methods were proposed to cope with the uneven-length problem for monitoring multiphase batch processes[35,36].

In this paper, a window-based step-wise sequential phase partition method is proposed for nonlinear batch processes with multiphase operations. By establishing the Kernel PCA (KPCA) models, the proposed method can effectively partition multiphase batch operations, such that existing monitoring methods like PCA can be efficiently applied for each phase. Moreover, a moving window is introduced to analyze the process dynamics. The three-dimensional (3D) batch data are unfolded in the batch-wise direction, such that the storage space and the computation complexity for kernel matrices can be significantly reduced. Note that the proposed phase partition method takes into account the process nonlinearity, dynamics and time sequence simultaneously, therefore capable of improving phase partition accuracy and reliability. Moreover,

a window-based step-wise sequential phase partition algorithm is specifically developed for uneven-length batch processes.

For clarity, the paper is organized as follows: Section 2 presents the problem description and some preliminary knowledge of KPCA, along with two monitoring indices. The proposed phase partition algorithms are detailed in Section 3 for even- and uneven-length batch processes, respectively. In Section 4, a numerical case and two industrial multiphase processes, an injection molding process and a penicillin fermentation process, are given to demonstrate the effectiveness and merit of the proposed method. Finally, some conclusions were drawn in Section 5.

2 Problem description and preliminaries

2.1 Multiphase process monitoring

Generally, the data collected from a batch process can be arranged into a 3D matrix denoted by $\boldsymbol{X}\left(I\times J\times K\right)$, where I is the batch index, J the number of variables and K the number of sample points. Correspondingly, denote $\boldsymbol{X}_k\left(I\times J\right)$, $k=1,\ldots,K$, the time-slice matrix for analysis. Owing to the fact that the process features are related to the operation conditions rather than the individual sample points, the correlation of process variables has similarity within certain time intervals, termed phase for batch process operation. Since the variable correlation changes from one phase to another, the process features are different over phases, denoted by

$$f_1\left(\boldsymbol{X}_1,\cdots,\boldsymbol{X}_{k_1}\right),\ f_2\left(\boldsymbol{X}_{k_1+1},\cdots,\boldsymbol{X}_{k_1+k_2}\right),\cdots,\ f_d\left(\boldsymbol{X}_{k_1+k_2+\cdots+k_{d-1}+1},\cdots,\boldsymbol{X}_K\right) \tag{1}$$

where $\boldsymbol{X}_k$ $(k=1,\cdots,K)$ denotes the time-slice matrix, $f_i\left(i=1,\cdots,d\right)$ is a linear or nonlinear mapping function, corresponding to the loading matrix in PCA for a linear process, $k_i\left(i=1,\cdots,d\right)$ is the number of time-slice matrices in the i-th phase, and d is the number of phases.

For a nonlinear batch process, using the KPCA approach to (1) gives

$$f_1\left(\Phi\left(\boldsymbol{X}_1\right),\cdots,\Phi\left(\boldsymbol{X}_{k_1}\right)\right),\ \cdots,\ f_d\left(\Phi\left(\boldsymbol{X}_{k_1+k_2+\cdots+k_{d-1}+1}\right),\cdots,\Phi\left(\boldsymbol{X}_K\right)\right) \tag{2}$$

where Φ is a nonlinear function, mapping the original data into a high-dimensional space.

If the process can be divided into different phases, the classical strategy of using a single model for each phase can be implemented for process monitoring. Proper partition of the process

into multiple phases is therefore key important for statistical modeling and monitoring. The vital problem lies with how to determine the number of phases properly. To solve the problem for batch processes with multiphase operations, a window-based step-wise sequential phase partition algorithm is proposed herein, which takes the process time sequence and dynamics into consideration simultaneously. For the ease of understanding, some preliminary knowledge is given firstly in the next subsection.

2.2 Preliminaries

2.2.1 Kernel PCA

The basic idea behind KPCA is to map the input space into a high-dimensional space (feature space) via a nonlinear map, and subsequently extract the principal components in the feature space. An important merit of the kernel based method is that the exact expression of the nonlinear map needs not to be figured out by the user, owing to that the computation can be transformed into the inner product of nonlinear mapping functions in the feature space, which therefore is called 'kernel trick'.

For the use of KPCA, the sampled data $\boldsymbol{x}_i \in \Re^m, i = 1, \ldots, N$, is firstly mapped into a feature space *F*, where *N* is the number of samples. Then the principal components are obtained by solving the eigenvalues of the covariance matrix in the feature space,

$$\begin{cases} \lambda \boldsymbol{v} = C^F \boldsymbol{v} \\ \boldsymbol{C}^F = \dfrac{1}{N} \sum_{i=1}^{N} \Phi(\boldsymbol{x}_i) \Phi(\boldsymbol{x}_i)^T \end{cases} \tag{3}$$

where $\Phi(\cdot)$ is a nonlinear mapping function, assuming that $\Phi(\boldsymbol{x}_i)$, $i = 1, \ldots, N$ is mean centered and variance scaled, eigenvalue $\lambda \geq 0$ and eigenvector $\boldsymbol{v} \in F \setminus \{0\}$.

Note that $\boldsymbol{C}^F \boldsymbol{v}$ can be written as

$$\boldsymbol{C}^F \boldsymbol{v} = \frac{1}{N} \sum_{i=1}^{N} \langle \Phi(\boldsymbol{x}_i), \boldsymbol{v} \rangle \Phi(\boldsymbol{x}_i) \tag{4}$$

where $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is the inner product between $\boldsymbol{x}$ and $\boldsymbol{y}$. This implies that all solutions of $\boldsymbol{v}$ with $\lambda \neq 0$ must lie in the span of $\Phi(\boldsymbol{x}_1), \ldots, \Phi(\boldsymbol{x}_N)$, that is, there exists nonzero coefficients β_i

$(i = 1, \ldots, N)$ such that

$$\boldsymbol{v} = \sum_{i=1}^{N} \beta_i \Phi(\boldsymbol{x}_i) \tag{5}$$

Substituting (4) and (5) into (3), and defining an $N \times N$ matrix $\boldsymbol{K}$ by $\boldsymbol{K}_{ij} = \left\langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \right\rangle$, one obtains

$$N\lambda\boldsymbol{\beta} = \boldsymbol{K}\boldsymbol{\beta} \tag{6}$$

where $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_N]^T \in \Re^{N \times 1}$. Then by solving the eigenvalues of (6), the eigenvectors $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n$ are derived with respect to the nonzero eigenvalues, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$. Note that $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n$ should be normalized by requiring the corresponding vectors in *F* to be normalized, i.e.

$$\left\langle \boldsymbol{v}_j, \boldsymbol{v}_j \right\rangle = 1, \text{ for } \quad j = 1, \ldots, n. \tag{7}$$

According to (5), there is $\lambda_j \left\langle \boldsymbol{\beta}_j, \boldsymbol{\beta}_j \right\rangle = 1$. The scores of a test vector $\boldsymbol{x}$, denoted by $\boldsymbol{t}$, are then extracted by projecting $\Phi(\boldsymbol{x})$ onto the eigenvectors $\boldsymbol{v}_j$ of *F*,

$$t_j = \left\langle \boldsymbol{v}_j, \Phi(\boldsymbol{x}) \right\rangle = \sum_{i=1}^{N} \beta_{i,j} \left\langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}) \right\rangle = \sum_{i=1}^{N} \beta_{i,j} \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}), \; j = 1, \ldots, r. \tag{8}$$

where *r* is the number of retained principal components (PCs), $\beta_{i,j}$ is the *i*-th element in vector β_j, and $\boldsymbol{t} = [t_1, \cdots, t_r] \in \Re^{1 \times r}$.

Before applying KPCA, mean centering and variance scaling should be performed in the feature space. Mean centering is made by

$$\boldsymbol{K} = \boldsymbol{K} - \boldsymbol{1}_N \boldsymbol{K} - \boldsymbol{K}\boldsymbol{1}_N + \boldsymbol{1}_N \boldsymbol{K}\boldsymbol{1}_N \tag{9}$$

where

$$\boldsymbol{1}_N = \frac{1}{N} \begin{bmatrix} 1 & \cdots & 1 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ 1 & \cdots & 1 \end{bmatrix} \in \Re^{N \times N}$$

Variance scaling is made by

$$\boldsymbol{K}_{scl} = \frac{\boldsymbol{K}}{\text{trace}(\boldsymbol{K}) / (N-1)} \tag{10}$$

The detailed derivation of the above formulae can be found in the reference[21] and thus is omitted. For simplicity, the commonly used radial basis kernel in the literature, $k(\boldsymbol{x},\boldsymbol{y})=\exp\left(-\|\boldsymbol{x}-\boldsymbol{y}\|^2/\gamma\right)$, is taken as the kernel function, where γ is a user specified parameter.

2.2.2 Hotelling's T^2 statistic and the Q-statistic

Similar to PCA, the sum of the normalized squared scores named Hotelling's T^2 statistic, and the squared prediction error (SPE) also named the *Q*-statistic, are used in KPCA monitoring methods, defined, respectively, by

$$T^2=[t_1,...,t_r]\Lambda^{-1}[t_1,...,t_r]^T \tag{11}$$

$$SPE=\left\|\Phi(x)-\Phi(x)\right\|^2=\sum_{j=1}^{n}t_j^2-\sum_{j=1}^{r}t_j^2 \tag{12}$$

where *n* is the number of nonzero eigenvalues and these eigenvalues are determined from (6), *r* is the number of the retained PCs, t_j is computed from (8), and Λ is a diagonal matrix of the eigenvalues corresponding to the retained PCs.

The confidence limit for T^2 is computed using the *F*-distribution:

$$T^2_{r,N,\mu}\approx\frac{r(N-1)}{N-r}F_{r,N-r,\mu} \tag{13}$$

where *N* is the number of samples, the subscript μ is the significance level.

The confidence limit for the SPE can be computed from its distribution approximation,

$$\mathrm{SPE}_{\mu}\approx g\chi^2_{\tau,\mu} \tag{14}$$

where $g=b/2a$, $\tau=2a^2/b$, *a* and *b* are the estimated mean and variance of the SPE.

Generally, the SPE index can reflect the disruption of the normal process data correlation that corresponds to an abnormal operation. Note that for the case of a sample exceeding the T^2 limit rather than the SPE limit, it perhaps drifts away from the origin in the PCs, and therefore could be a permitted status during the process operation rather than a fault[37]. Hence, the SPE index is adopted in this study for efficiently monitoring batch processes with multiphase operations, including fault detection.

3 Proposed multiphase partition

In this work, batch processes are first partitioned by the proposed method, then each phase-based monitoring models are established separately. The framework of this paper is given in Figure 1. Phase partition methods are the main works in this paper. In order to get better partition performance, moving window method is modified in our work. Considering the different characters existed in even- and uneven-length batch processes, the proposed method are presented in detail for the two cases separately in the following subsection.

3.1 Phase partition algorithm for even-length batch processes

With the 3D matrix data collected from a batch process denoted by $\boldsymbol{X}(I \times J \times K)$, it is necessary to unfold the 3D matrix into a 2D matrix for the convenience of analysis. The batch-unfolding $\left(\boldsymbol{X}_b(I \times KJ)\right)$ and variable-unfolding $\left(\boldsymbol{X}_v(KI \times J)\right)$ strategies are illustrated in Figure 1, where the batch dimension and the variable dimension remain unchanged, respectively. The variable-wise unfolding strategy can reflect the variation along the time rather than the batch-wise direction. In contrast, the batch-wise unfolding strategy for modeling facilitates observing the batch-wise variation throughout the cyclic operation of a batch process. In this work, the batch-wise unfolding strategy is preferred to establish KPCA models for phase partition. Note that the storage space and computation effort can be significantly reduced compared to the use of a variable-wise unfolding method. The reason is interpreted as below.

Denote by w the moving window length for phase partition, the modeling matrix is $\boldsymbol{X}(I \times wJ)$ if unfolded in the batch-wise direction, and correspondingly, the kernel matrix and loading matrix are in the forms of $\boldsymbol{K}(I \times I)$ and $\boldsymbol{P}(I \times n_1)$, respectively. In contrast, if unfolded in the variable-wise, the modeling matrix is $\boldsymbol{X}(wI \times J)$ and the corresponding kernel matrix and loading matrix are $\boldsymbol{K}(wI \times wI)$ and $\boldsymbol{P}(wI \times n_2)$. Note that n_1 and n_2 are the corresponding number of nonzero eigenvalues, respectively. It is obvious that the storage space and computation effort for these batch-wise unfolded matrices are much less than those of the variable-wise unfolding strategy.

In the sequel, batch-wise normalization is adopted to ensure that all the time-slice matrices $\boldsymbol{X}_k(I \times J)$, $k = 1, \ldots, K$ have zero mean and unit variance. Then perform KPCA on each

normalized time-slice matrix, i.e. computing the kernel matrix $\boldsymbol{K}_k$ based on a choice of the kernel function. The PCs retained for each model and the unified number of PCs are determined by

$$\frac{\sum_{j=1}^{r_k}\lambda_j^k}{\sum_{j=1}^{n_k}\lambda_j^k} > \eta\%,\ k=1,2,\cdots,K \tag{15}$$

where η indicates the percentage of data information retained in the principal component space. The confidence limit of SPE_k ($k=1,2,\cdots,K$), denoted by Ctr_k, is computed using (12) and (14) based on the retained PCs for each model.

Owing to using a window based matrix $\boldsymbol{X}_w(I\times wJ)$ to establish a phase partition algorithm, the column dimension of the modeling matrix is kept invariant, which facilitates determining the kernel parameter and the use of KPCA. Another merit is that the time sequence of these time-slice matrices can be preserved. In order to capture the process transition between phases, only one time step is taken to move forward these time-slice matrices in a window for computation. A window based KPCA modeling procedure is given as follows.

Firstly, compute the kernel matrix $\boldsymbol{K}_w(I\times I)$ of the window based matrix $\boldsymbol{X}_w(I\times wJ)$, and normalize $\boldsymbol{K}_w(I\times I)$ using (9) and (10). Secondly, solve the corresponding eigenvalues from (6) to establish the loading matrix $\boldsymbol{P}_w(I\times n)$ by retaining those eigenvectors with nonzero eigenvalues aligned in a descending order. Thirdly, calculate the kernel matrix $\boldsymbol{K}_k(I\times I)$ of the time-slice matrix $\boldsymbol{X}_k(I\times J)$ in terms of a moving window. Each kernel matrix is normalized using the window information in terms of (9) and (10) as

$$\boldsymbol{K}_k = \boldsymbol{K}_k - \boldsymbol{1}_I\boldsymbol{K}_w - \boldsymbol{K}_k\boldsymbol{1}_I + \boldsymbol{1}_I\boldsymbol{K}_w\boldsymbol{1}_I \tag{16}$$

$$\boldsymbol{K}_{k,scl} = \frac{\boldsymbol{K}_k}{\text{trace}(\boldsymbol{K}_w)/(I-1)} \tag{17}$$

where $\boldsymbol{1}_I$ is defined as below

$$\mathbf{1}_I = \frac{1}{I}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} \in \Re^{I\times I}$$

Fourthly, project each normalized kernel matrix $\boldsymbol{K}_{k,scl}$ on $\boldsymbol{P}_w$ to calculate the SPE value. Given a vector $\boldsymbol{x}$ in the time-slice matrix $\boldsymbol{X}_k\left(I\times J\right)$ within the window, it can be computed that

$$t_{k,j} = \sum_{i=1}^{I} \beta_{i,j} \boldsymbol{K}_{k,scl}(\boldsymbol{x}_i, \boldsymbol{x}) \tag{18}$$

$$SPE_{w,k} = \sum_{j=1}^{n} t_{k,j}^2 - \sum_{j=1}^{r} t_{k,j}^2 \tag{19}$$

where $\beta_{i,j}$ is the element at the *i*-th row and the *j*-th column of the window based loading matrix $\boldsymbol{P}_w\left(I\times n\right)$, and the subscript *k* is the index of the used time-slice matrix. Accordingly, the confidence limit $Ctr_{w,k}$ of $SPE_{w,k}$ is computed using (14).

Compare $Ctr_{w,k}$ with Ctr_k for each time-slice matrix in the window to find the time k^* that satisfies the condition for three consecutive samples,

$$Ctr_{w,k} > \alpha Ctr_k \tag{20}$$

where α is a tuning parameter, termed relaxing factor, which determines the loss tolerance of reconstruction power of window-based KPCA model in comparison with the associated time-slice KPCA models. Equation (20) means that the accuracy of window-based KPCA model is thus significantly worse than that of time-slice KPCA models. So α determines how much the window-based KPCA model is allowed to be less representative than time-slice KPCA models. The matrix index before k^* is determined as one phase. Generally, it is suggested to use $SPE_{w,k}$ of the first window for computation to determine $Ctr_{w,k}$, and correspondingly, the tuning range of α is preferred to be between $\min\left(Ctr_{w,k}/Ctr_k\right)$ and $\max\left(Ctr_{w,k}/Ctr_k\right)$. An initial value of α may be taken as the mean of $Ctr_{w,k}/Ctr_k$ in this range. In fact, different partition results may be obtained by different choices of α. A larger value of α means the use of less monitoring models; on the contrary, a smaller α means the use of more accurate monitoring models to describe each time slice. Therefore, the choice of α is the compromise between

model accuracy and complexity. The best partition result may be obtained if a prior knowledge of the process could be used for the choice of α. When there is no prior knowledge of the process phase information, the best partition may be determined by choosing α to optimize the monitoring performance for historical process data with known fault information. Note that a similar algorithm can be established in terms of the T^2 statistic index. However, it may not be sensitive as well as the use of SPE to distinguish underlying variations of process characters[19]. Therefore, the *Q*-statistic index is adopted herein. For clarity, the proposed multiphase partition algorithm for even-length batch processes named WNSSPP-E is summarized as below. The corresponding flow diagram is shown in Figure 2.

Summary of the proposed WNSSPP-E algorithm

Step 1: Unfold $\boldsymbol{X}(I \times J \times K)$ into $\boldsymbol{X}_k(I \times J)$ in batch-wise, followed by normalization to be zero mean and unit variance.

Step 2: Perform KPCA on each $\boldsymbol{X}_k(I \times J)$ using (6)-(10), along with the computation of SPE and the confidence limit Ctr_k using (12) and (14).

Step 3: By establishing a window based KPCA model using (6)-(10) based on the batch-wise unfolded matrix $\boldsymbol{X}_w(I \times wJ)$, compute $SPE_{w,k}$ for the last three time-slice matrices within the window using (16)-(19) and similarly determine the confidence limits $Ctr_{w,k}$ using (14).

Step 4: Compare $Ctr_{w,k}$ with Ctr_k to find the time k^* that satisfies for sequential three samples the condition in (20). The matrix index before k^* is determined as one phase.

Step 5: If k^* is found, take out the determined phase and then proceed with the remaining process data for phase partition. Otherwise, one time step is taken to move forward the time-slice matrices in the window and return to steps 3 and 4 till the end of data length.

3.2 Phase partition algorithm for uneven-length batch processes

Figure 3 gives an illustration of uneven-length batch processes, where the process phases are

assumed to be two, one is indicated by the blank area and the other by the shaded area. Four different cases of uneven-length batch operation are shown, where A and C denote the time intervals that all batch data belong to the first and second phases, respectively; B and D denote the time intervals that part of different batch data belong to the first or second phases, respectively. It is seen that measurements on the same sampling interval in different batches may belong to different phases, causing the standard batch-wise normalization meaningless. To deal with the uneven-length problem, another phase partition algorithm is proposed as below.

For uneven-length batch processes, it makes sense to assign the initial samples of each batch into the same phase. Correspondingly, the initial w time-slice matrices (i.e. time-slice matrices in the first moving window), denoted by $\boldsymbol{X}_{w1}\left(I\times wJ\right)$, are normalized to be zero mean and unit variance. Then a KPCA model is established based on the normalized matrix, $\boldsymbol{X}_{w1}\left(I\times wJ\right)$, to obtain the initial training model. The initial training model is used to check the subsequent samples in each batch. Taking the *k*-th sample in the *i*-th batch $\boldsymbol{x}_{i,k}(1\times J)$ for example, the sample is augmented by adding in the previous samples within a window length to be $\boldsymbol{x}_{i,k}^{w}\left(\boldsymbol{x}_{i,k-w+1}(1\times J),\cdots,\boldsymbol{x}_{i,k}(1\times J)\right)$. In this way, the column dimension of the augmented sample is the same as that of the modeling matrix, and the Gram vector $\boldsymbol{\theta}$ can be computed using a specified kernel function in the feature space, i.e.

$$\boldsymbol{\theta}=\mathrm{K}(\boldsymbol{x}_{i,j}^{w},\boldsymbol{X}_{w1}) \tag{21}$$

Then $\boldsymbol{\theta}$ is projected on the initial KPCA model for computing the SPE index. If three consecutive samples thus augmented are beyond the relaxed control limit, αCtr_{w1}, the last sample before them is taken as a phase partition point. Otherwise, these samples are counted into the current phase, and the modeling matrix is updated by substituting $\boldsymbol{x}_{i,k}^{w}\left(\boldsymbol{x}_{i,k-w+1}(1\times J),\cdots,\boldsymbol{x}_{i,k}(1\times J)\right)$ with $\boldsymbol{x}_{i,k+1}^{w}\left(\boldsymbol{x}_{i,k-w+2}(1\times J),\cdots,\boldsymbol{x}_{i,k+1}(1\times J)\right)$. In this way, the column dimension of modeling matrix in each step is kept invariant while all the samples in the modeling matrix belong to the current phase. Due to the phases are varied over batches, the phase partition points will be different from batch to batch. Therefore, it is proposed to find out all the first phases of each batch, and then remove them to continue the determination of the second phase of all batches as before, until all phases are found for each batch. To facilitate

understanding, we give a graphical illustration of the updating strategy for the ith batch of the proposed WNSSPP-U algorithm in Figure 4.

Compared to the WNSSPP-E algorithm for even-length batch processes, The algorithm for uneven-length batch processes, named WNSSPP-U, are different in that: 1. the partition points are not simultaneously determined for all batches; 2. The SPE values of three consecutive samples as above augmented are compared with the control limit computed from the modeling matrix to judge whether these samples belong to the current phase or not in WNSSPP-U algorithm. In contrast, the control limits are compared to determine the partition points for even-length batch processes.

For clarity, the proposed algorithm for uneven-length batch processes is summarized as below. The corresponding flow diagram is shown in Figure 5.

Summary of the proposed WNSSPP-U algorithm

Step 1: Normalize the first *w* time-slice matrices, denoted by $\boldsymbol{X}_w\left(I \times wJ\right)$, to be zero mean and unit variance.

Step 2: Perform KPCA on $\boldsymbol{X}_w\left(I \times wJ\right)$ using (6)-(10) and compute the confidence limit using (12) and (14).

Step 3: Project the augmented samples on the established KPCA model and calculate the corresponding SPE values for each batch to determine the partition points. The modeling matrix is updated by moving forward the window by one sample.

Step 4: Repeat step 1-3 till all partition points for all batches are determined. Remove the data belonging to the previously determined phase, and then use the remaining data for further partition. Repeat step 1-4 till all samples are checked over.

3.3 Choice of the kernel parameter and window width

Since the kernel parameter γ in a radial basis kernel affects the modeling results, it was suggested to take $\gamma = 5J$ for the use of KPCA[22,24], where J is the dimension of sample variables. However, it is found by simulation tests that the numerical choice cannot in general

guarantee good results for various processes. A general guideline is therefore suggested for practical application, where γ is an adjustable parameter to be determined. In consideration of that the number of the retained PCs also affects the partition results, these parameters of η, w and γ should be determined together.

The ranges of these three parameters are $\eta \in \{85, 90, 95\}$, $\gamma \in (0, \gamma_{\max})$, and $w \in (0, w_{\max})$, e.g. $\gamma_{\max} = 100$ and $w_{\max} = 30$. A cost function is important to determine the optimal parameters, which is generally defined as the mean-squared prediction error[38]. If the real partition points could be known with a prior knowledge of the process, the following cost function may be used,

$$\text{cost}(w, \gamma) = \sum_i |s_i - \hat{s}_i| \tag{22}$$

where s_i denotes real partition points, $\hat{s}_i$ indicates the predicted partition points.

The optimal choice of w, γ, and η can be determined by numerically sweeping over the possible range of these parameters for computing the minimum of $\text{cost}(w, \gamma)$. For computation efficiency, it is suggested to take the searching step of $l_1 = 1$ for $w \in (0, w_{\max})$, $l_2 = 1$ for $\gamma \in (0, \gamma_{\max})$, and $l_3 = 5$ for $\eta \in \{85, 90, 95\}$. A traversal algorithm is given below.

Summary of the traversal algorithm

---

1. Initialize $\eta = 85$, i=1, $w = 1$, $\gamma = 0$, $C_\gamma = M$, and $C_{w,\gamma} = M$, where M is a large natural number. e.g. $M = 10^6$;
2. Let $\gamma = \gamma + l_2$, establish a KPCA model using (6)-(10). By choosing α in a range of $[\min(Ctr_{w,k} / Ctr_k), \max(Ctr_{w,k} / Ctr_k)]$ determined using $SPE_{w,k}$ of the first window for computation, different cost function $\text{cost}(w, \gamma)$ can be computed using (22). Denote by $\text{cost}_m(w, \gamma)$ the minimum.
3. If $\text{cost}_m(w, \gamma) \le C_\gamma$, let $C_\gamma = \text{cost}_m(w, \gamma)$ and return to step 2 for $\gamma \le \gamma_{\max}$. Otherwise, return to step 2 for $\gamma \le \gamma_{\max}$. If $\gamma > \gamma_{\max}$, by letting $\gamma = 0$, go to the next step;
4. Let $w = w + l_1$. If $C_\gamma \le C_{w,\gamma}$, let $C_{w,\gamma} = C_\gamma$ and return to step 2 for $w \le w_{\max}$. Otherwise, return to step 2 for $w \le w_{\max}$. If $w > w_{\max}$, by letting $C_i = C_{w,\gamma}$, $\gamma = 0$ and $w = 5$, go to the next step;
5. If $i < 3$, let $i = i + 1$ and $\eta = \eta + 5$, return to step 2. Otherwise, go to the next step;

6. The optimal w, γ, and η are determined corresponding to the minimum of C_i.

3.4 Online monitoring

The proposed phase partition algorithms are established offline based on the historical batch data. The resulting phases facilitate establishing effective monitoring models. After proper phase partition, phase-based monitoring models can be established for online monitoring. Note that if batch-wise unfolding is used for establishing online monitoring models, the future samples will be required for estimation, which will definitely degrade the monitoring performance. Hence, variable-wise unfolding is adopted to establish the monitoring models. Although both KPCA and PCA methods can be used to establish online models, the computation effort of using KPCA is relatively higher compared to a PCA modeling method. For the convenience of practical applications, the variable-wise unfolding strategy and PCA modeling are preferred for monitoring and fault detection. That is, the samples in each divided phase are unfolded in the variable-wise direction and normalized. Then a PCA model is established based on the normalized data. Finally, the control limits are computed for online monitoring. The PCA algorithm for online monitoring can be found in the literature and therefore is omitted.

4 Illustrative Examples

A numerical example and two multiphase batch processes, an injection molding (IM) process and a fed-batch penicillin fermentation process, are used to demonstrate the effectiveness of the proposed multiphase partition method.

Example 1. Consider a three-phase polynomial numerical example described by

$$\begin{cases} x_1 = 2t \\ x_2 = t^2 + 3t - 1 \\ x_3 = -t^3 + 3t^2 \end{cases} \quad \begin{cases} x_1 = t^2 + 2t \\ x_2 = t^3 + t^2 \\ x_3 = 3t \end{cases} \quad \begin{cases} x_1 = t^4 - t^2 + 2 \\ x_2 = 3t - 1 \\ x_3 = 2t^2 - 1 \end{cases} \tag{23}$$

where t is uniformly distributed in a range of $[0.01, 2]$. For illustration, thirty samples are generated for each of the phases using (23) and the generated data are corrupted with white noise of $N(0, 0.01)$.

In this case, the parameters w, γ, and η are determined using the cost function (22) in the traversal algorithm. For the use of the proposed WNSSPP-E algorithm in comparison with the SSPP algorithm developed for multiphase batch processes, the retained PCs should explain at least 90% of the data variation and the significance level is taken as $\mu = 99\%$, together with for computation. For using the SSPP algorithm, each time slice matrix is normalized to be zero mean and unit variance, and perform PCA on each time slice matrix to get the initial model and control limits. Then the following PCA models with respect to each time segment is established by adding the sequential time slice matrix via the variable-wise unfolding, such that the partition points are determined. The detail of the SSPP algorithm can be found in the reference[19].

The partition results from SSPP and WNSSPP-E are plotted in Figures 6 and 7, respectively, where the vertical dash lines indicate the true partition points. Figure 6 shows that there are large deviations between the real partition points and the predicted ones by using the SSPP algorithm, although different choices of α are taken for computation. By setting $w = 6$ and $\gamma = 27$, it is seen from Figure 7 that the proposed WNSSPP-E method can give almost exact partition result when taking $\alpha = 2.96$, except for two samples time lag between phase, in contrast with the real partition points. It is obvious that the nonlinear feature of the numerical example is better captured by the proposed WNSSPP-E method. The partition performance of the proposed WNSSPP-E method based on variable-wise unfolding are shown in Figure 8 for comparison. It is seen that the three phases cannot be exactly divided with regard to different choices of α. Note that a smaller or larger value of α will give worse partition results.

To demonstrate the performance of the proposed WNSSPP-U algorithm for uneven-length batch processes, it is assumed that 25 samples are generated in the first phase for the first 10 batches, 30 samples for the subsequent 10 batches, and 35 samples for the last 10 batches. The other two phases include 30 samples as above for all batches. By taking $w = 1$, $\gamma = 21$ and $\alpha = 3.7$ in the proposed WNSSPP-U algorithm, the partition results for obtaining the first phase are shown in Figure 9. It can be seen that the first phases of these uneven-length batches are accurately divided.

Example 2. Consider an IM process studied in the references[17,39]. The IM process is

comprised of four main phases, filling, packing-holding, plastication and cooling. The detailed phase description were given in these references. Here, both even- and uneven-length IM processes are involved.

For the even-length IM process, by deleting the meaningless data at the end of each batch, a historical process database including $\boldsymbol{X}(50\times11\times500)$ is used here. The measured variables in the injection molding process are shown in Table 1. By unfolding the 3D matrix into batch-wise 2D matrices, KPCA is performed on each normalized time-slice matrix, $\boldsymbol{X}_k(50\times11)$. According to the process operation knowledge, the real partition points of the four main phases are specified as 64, 117, 275 and 368 to evaluate the phase partition performance.

The partition results by using the SSPP algorithm are shown in Figure 10, where the four vertical dash lines indicate the true transition points of the main process phases. When $\alpha=1.5$, four stable phases are divided. However, the transitions are mostly detected before the occurrence of the true phase transitions by the SSPP algorithm. Using the proposed WNSSPP-E algorithm with the window length taken as $w=20$ and $\gamma=3$, the partition results are shown in Figure 10. Taking $\alpha=1.82$ for example, the proposed WNSSPP-E algorithm gives almost correct partition results. Moreover, it can be seen that the partition points mostly occur right after the real transitions happened, though certain time delay appears to a few partition points. That is to say, there are seldom erratic partitions by using the proposed WNSSPP-E algorithm. Comparison of the partition results by taking $\alpha=1.5$ of SSPP and $\alpha=1.82$ of the proposed WNSSPP-E algorithm is given in Table 2, where the partition points before real transition points are denoted as error (marked in red). It is seen that incorrect partitions are evidently reduced by using the proposed WNSSPP-E algorithm compared to the SSPP algorithm. The partition performance of the proposed WNSSPP-E algorithm based on variable-wise unfolding are shown in Figure 12 for comparison. It is seen that the partition results are apparently inferior than those shown in Figure 10, despite different choices of α.

To demonstrate the effectiveness of the proposed method for tackling the uneven-length problem of IM process operation, the injection velocity is changed from 22 to 26 mm/s, including three typical velocity values: 22, 24, 26 mm/s. By eliminating the data collected before injection,

there are 81 to 94 samples in the remaining injection phase, where 81 samples consist of the minimal filling duration corresponding to the injection velocity of 26 mm/s, and 94 samples consist of the maximal injection phase duration corresponding to injection velocity 22 mm/s. Totally, 23 batches are used for modeling. The first 7 batches correspond to the injection velocity of 26 mm/s, the subsequent 8 batches corresponding to the injection velocity of 24 mm/s, and the last 8 batches corresponding to the injection velocity of 22 mm/s. Using the proposed WNSSPP-U algorithm with a window length of $w = 3$, the partition results are shown in Figures 13 and 14. In Figure 13, it is seen that four phases of these batches with different injection velocities are correctly divided by taking $\alpha = 1.9$. The partition points determined for each batch based on the historical batch data are compared with the real partition points in Figure 14, well demonstrating the effectiveness of the proposed method.

Example 3. Consider the penicillin fermentation process studied in the literature[40,41]. A modular simulator named PenSim 2.0 for the penicillin fed-batch fermentation had been developed by the monitoring and control group of the Illinois Institute of Technology in 2002. Generally, the penicillin fermentation process is divided into four phases, including the lag phase, the exponential growth phase, the stationary phase, and the autolysis phase. Also, it can be divided into two operation stages, the pre-culture stage (including the lag phase and the exponential growth phase) and the fed-batch stage (including the stationary phase and the autolysis phase).

In this study, 50 normal batches data are generated from PenSim 2.0, where 11 variables are considered as listed in Table 3. The typical time-profiles of nine variables are shown in Figure 15. The duration of each batch is 400 hours, about 45 hours for the pre-culture stage and the rest about 355 hours for the fed-batch stage. The starting point of the fed-batch stage can be used to estimate the partition performance for different choice of α. The sampling time for simulation is chosen to be 0.5 hours.

The partition results by using the SSPP algorithm are shown in Figure 16, where the vertical dash line indicates the fed-batch operation transition point in the penicillin fermentation process. When $\alpha = 1.55$, three stable phases are obtained, i.e. from the beginning to the sample point 38,

from the sample point 54 to 94, and the rest sample points. The small phases between the first and second stable phases are regarded as the transition phase. So, the process is divided into four phases, three stable phases and one transition phase. However, the fed-batch operation is detected with a notable delay. Moreover, the transition occurred earlier, which is indeed impossible from the process knowledge.

Using the proposed WNSSPP-E algorithm with a window length of $w = 5$ and $\gamma = 66$, the partition results are shown in Figure 16. It is seen that the true transition point of 90 to the fed-batch stage is exactly detected. When taking $\alpha = 4.3$, the process is divided into six phases. The first phase is from the beginning to the sample point 66, followed by some transition phases from the sample point 66 to 90. Note that these two phases are similar with the results given in the recent paper[29]. Then a small phase is divided after the fed-batch operation, which is reasonable because the added substrate begins to react in the fed-batch operation. Subsequently, the process enters into stable synthesis phases, from the sample point 185 to 345 and from 345 to 657. The autolysis phase is divided from the sample point 657 to the end. Hence, the partition result is coherent with the biological phases and operational transition of the fermentation process, which facilitates effectively monitoring the process.

To demonstrate the monitoring performance, 30 normal batches are used to establish the monitoring models and additional 15 batches are generated as the test set. Among the test set, there are 10 normal batches and 5 fault batches as described in Table 4. The variable-unfolding modeling method is adopted based on the proposed partition result. The batch data are unfolded in the variable-unfolding style and then PCA is performed on the rearranged data. The test data of normal process are well under control and do not give any alarm as shown in Figure 18, though some statistical indices are beyond the control limits. The monitoring results for fault 1 and fault 2 are given in Figure 19 and Figure 20, respectively. It is seen that the faults can be obviously detected. For comparison, the monitoring results by using a single model of PCA, the improved repeatability factor (IRF) method[29], KICA-PCA[16], and the SSPP algorithm[19] are listed in Table 5 with respect to the alarm time, where the red numbers indicate the best results obtained by using these methods. It is seen that the proposed method gives obviously earlier alarm time for faults

3-5 compared to the other methods. Moreover, the fault detection results by using these methods in terms of the SPE index are listed in Table 6. It is again seen that the proposed method gives obviously shorter detection delay for faults 2-5 compared to the other methods.

5 Conclusions

In this paper, a window-based step-wise sequential phase partition method has been proposed for nonlinear batch processes with multiphase operations. An important merit is that multiple phases, time sequence, nonlinearity, and process dynamics are simultaneously taken into accounts for practical applications. The 3D process data are proposed to be unfolded in the batch-wise direction for the convenience of computation. A moving window is introduced to improve the partition performance of KPCA. These two strategies facilitate capturing the process dynamic characteristics while saving the storage space of kernel matrices and computation effort. Moreover, for uneven-length batch processes, a specific partition algorithm has been developed to determine different phase points for each batch, so as to facilitate establishing monitoring models for each phase of the batch operation. The numerical example has well demonstrated the effectiveness and merit of the proposed method for even- or uneven-length batches. For the IM process, incorrect partitions can be significantly reduced by the proposed method. For the penicillin fermentation process, it is interesting to see that the partition result given by the proposed method is coherent with the process mechanism, and moreover, the monitoring results have shown that the faults can be evidently detected earlier by using the proposed partition results, well demonstrating the advantage for monitoring nonlinear batch processes.

Acknowledgement

This work is supported in part by the National Thousand Talents Program of China, NSF China Grants 61473054, the Fundamental Research Funds for the Central Universities of China, and EU FP7 (Ref: PIRSES-GA-2013-612230).

References

[1] Nomikos, P.; MacGregor, J. F. Monitoring batch processes using multiway principal component analysis. *AIChE Journal*. 1994, 40, 1361-1357.

[2] Nomikos, P.; MacGregor, J. F. Multi-way partial least squares in monitoring batch processes. *Chemom. Intell. Lab. Syst.* 1995, 30(1), 97-108.

[3] Chen, J. H.; Chen, H. H. On-line batch process monitoring using MHMT-based MPCA. *Chem. Eng. Sci.* 2006, 61, 3223-3239.

[4] Ferreira, A. P.; Lopes, J. A.; Menezes, J. C. Study of the application of multiway multivariate techniques to model data from an industrial fermentation process. *Anal. Chim. Acta.* 2007, 595, 120-127.

[5] Camacho, J.; Pico, J.; Ferrer, A. The best approaches in the online monitoring of batch processes based on PCA: does the modeling structure matter? *Anal. Chim. Acta.* 2009, 642, 59-69.

[6] Yao, Y.; Gao, F. R. A survey on multistage/multiphase statistical modeling methods for batch process. *Annual Reviews in Control* 2009, 33, 172-183.

[7] Ge, Z. Q.; Song, Z. H.; Gao, F. R. Review of recent research on data-based process monitoring. *Ind. Eng. Chem. Res.* 2013, 52(10), 3543-3562.

[8] Ündey, C.; Çinar, A. Statistical monitoring of multistage, multiphase batch processes. *IEEE Control Syst. Mag.* 2002, 22, 40-52.

[9] Ündey, C.; Ertunc, S.; Çinar, A. Online batch fed-batch process performance monitoring, quality prediction, and variable contribution analysis for diagnosis. *Ind. Eng. Chem. Res.* 2003, 42, 4645-4658.

[10] Camacho, J.; Pico′, J. Multi-phase principal component analysis for batch processes modeling. *Chemom. Intell. Lab. Syst.* 2006, 81, 127-136.

[11] Camacho, J.; Pico′, J. Online monitoring of batch processes using multi-phase principal component analysis. *J. Process Control* 2006, 16, 1021-1035.

[12] Camacho, J.; Pico′, J.; Ferrer, A. Multi-phase analysis framework for handling batch process data. *Journal of Chemometrics* 2008, 10.1002/cem.1151.

[13] Lu, N. Y.; Gao, F. R.; Wang, F. L. Sub-PCA modeling and on-line monitoring strategy for batch processes. *AIChE Journal*, 2004. 50, 255-259.

[14] Lu, N. Y.; Gao, F. R. Stage-based process analysis and quality prediction for batch processes. *Ind. Eng. Chem. Res.* 2005, 44, 3547-3555.

[15] Zhao, C. H.; Wang, F. L.; Lu, N.Y.; Jia, M. X. Stage-based soft-transition multiple PCA modeling and online monitoring strategy for batch processes. *J. Process Control* 2007, 17, 728-741.

[16] Zhao, C. H.; Gao, F. R.; Wang, F. L. Nonlinear batch process monitoring using phase-based kernel-independent component analysis principal component analysis (KICA-PCA). *Ind. Eng. Chem. Res.* 2009, 48, 9163-9174.

[17] Dong, W. W.; Yao, Y.; Gao, F. R. Phase analysis and identification method for multiphase batch processes with partitioning multi-way principal component analysis (MPCA) model. *Chinese Journal of Chemical Engineering*, 2012, 20(6) 1121-1127.

[18] Yao, Y.; Gao, F. R. Phase and transition based batch process modeling and online monitoring. *J Process Control*, 2009, 19, 816-826.

[19] Zhao, C. H.; Sun, Y. X. Step-wise sequential phase partition (SSPP) algorithm based statistical modeling and online process monitoring. *Chemom. Intell. Lab. Syst.* 2013, 125, 109-120.

[20] Zhao, C. H. Concurrent phase partition and between-mode statistical analysis for multimode and multiphase batch process monitoring. *AIChE Journal* 2014, 60, 559-573.

[21] Lee, J. M.; Yoo, C.; Lee, I. B. Fault detection of batch processes using multiway kernel principal component analysis. *Chemom. Intell. Lab. Syst.* 2004, 28, 1837-1847.

[22] Cho, J. H.; Lee, J. M.; Choi, S.W.; Lee, D.W.; Lee, I.B. Fault identification for process monitoring using kernel principal component analysis. *Chemical Engineering Science* 2005, 60, 279-288.

[23] Jia, M. X.; Chu, F.; Wang, F. L.; Wang W. On-line batch process monitoring using batch dynamic kernel principal component analysis. *Chemom. Intell. Lab. Syst.* 2010, 101, 110-122.

[24] Ge, Z. Q.; Yang, C. J.; Song, Z. H. Improved kernel PCA-based monitoring approach for nonlinear processes. *Chemical Engineering Science* 2009, 64, 2245-2255.

[25] Girolami, M. Mercer kernel-based clustering in feature space. *IEEE Transactions on Neural Networks*, 2002, 13, 780-784.

[26] Doan, X. T.; Srinivasan, R. Online monitoring of multi-phase batch processes using phase-based multivariate statistical process control. *Comput. Chem. Eng.* 2008, 32, 230-243.

[27] Maiti, S. K.; Srivastava, R. K.; Bhushan, M.; Wangikar, P. P. Realtime phase detection

based online monitoring of batch fermentation processes. *Process Biochem*. 2009, 44, 799-811.

[28] Ge, Z. Q.; Zhao, L. P.; Yao, Y.; Song, Z. H.; Gao, F. R. Utilizing transition information in online quality prediction of multiphase batch processes. *J. Process Control*, 2012 22, 599-611.

[29] Tang X. C.; Li, Y.; Xie, Z. Phase division and process monitoring for multiphase batch processes with transitions. *Chemom. Intell. Lab. Syst.* 2015, 145, 72-83.

[30] Ramaker, H.; Van Sprang, E.; Westerhuis, J.; Smilde, A. K. Dynamic time warping of spectroscopic batch data. *Analyt. Chim. Acta*. 2003, 498, 133 – 153.

[31] Giorgio, T.; Frans van den, B.; Claus, A. Correlation optimized warping and dynamic time warping as preprocessing methods for chromatographic data. *J. Chemometr*. 2004, 18, 231-241.

[32] Srinivasan, R.; Qian, M. S. Online temporal signal comparison using singular points augmented time warping. *Ind. Eng. Chem. Res*. 2007, 46, 4531−4548.

[33] Zhao, L. P.; Zhao, C. H.; Gao, F. R. Inner-phase analysis based statistical modeling and online monitoring for uneven multiphase batch processes. *Ind. Eng. Chem. Res*. 2013, 52, 4586-4596.

[34] Li W. Q.; Zhao C. H.; Gao F. R. Sequential time slice alignment based unequal-length phase identification and modeling for fault detection of irregular batches. *Ind. Eng. Chem. Res.* 2015, 54, 10020-10030.

[35] Yu, J.; Qin, S. J. Multiway Gaussian mixture model based multiphase batch process monitoring. *Ind. Eng. Chem. Res*. 2009, 48, 8585–8594.

[36] Yao, Y.; Dong, W. W.; Zhao, L. P.; Gao, F. R. Multivariate statistical monitoring of multiphase batch processes with between-phase transitions and uneven operation durations. *Can J. Chem Eng*. 2012; 90:1383–1392.

[37] Qin, S. J. Statistical process monitoring: basics and beyond. *J. Chemometrics* 2003, 17, 480-502.

[38] Zoltán, B.; László, D.; János, A. Dynamic principal component analysis in multivariate time-series segmentation. *Conservation, Information, Evolution* 2011, 1, 11-24.

[39] Yang, Y. Injection molding control: from process to quality. Ph.D. thesis, Hong Kong University of Science and Technology, HongKong, 2004, 2-3.

[40] Birol, G.; Ündey, C.; Çinar, A. A modular simulation pack-age for fed-batch fermentation:

penicillin production. *Comput. Chem. Eng.* 2002, 26, 1553-1565.

[41] Hong, J. J.; Zhang, J.; Morris, J. Fault localization in batch processes through progressive principal component analysis. *Ind. Eng. Chem. Res.* 2011, 50, 8153-8162.

List of Table and Figure Captions

Table 1. The variable set adopted for monitoring an injection molding process

Number	Variables
1	Valve opening 1
2	Valve opening 2
3	Screw stroke
4	Screw velocity
5	Ejector stroke
6	Mold stroke
7	Mold velocity
8	Injection press
9	Barrel temperature 1
10	Barrel temperature 2
11	Barrel temperature 3

Table 2. Comparison of partition results for an IM process by using different algorithms

Method		Filling	Packing-holding	Plastication	Cooling
True phase		64-117	117-275	275-368	368-500
	partition	61-116	116-255	267-347	363-500
SSPP	error	3 1	1 20	8 21	5 0
	delay	- -	- -	- -	- -
	partition	67-118	118-270	272-370	377-500
WNSSPP	error	- -	- 5	3 -	- -
	delay	2 1	1 -	0 2	- 0

Table 3. The variable set adopted for monitoring a penicillin fermentation process

Number	Variables
1	Aeration rate(1 h^{-1})
2	Agitator power(W)
3	Substrate feed rate(1h^{-1})
4	Substrate feed temperature(K)
5	Dissolved oxygen concentration(%saturation)
6	Culture volume(L)
7	Carbon dioxide concentration(mmol/L)
8	PH
9	Bioreactor temperature(K)
10	Generated heat
11	Cooling water flow rate(1h^{-1})

Table 4. Description of five faulty batches of a penicillin fermentation process

No.	Variable	Fault type	Fault magnitude	Staring time	Terminal time
1	Agitation power	Step	-10%	30	200
2	Agitation power	Ramp	-1	50	200
3	Substrate feed rate	Step	10%	100	200
4	Substrate feed rate	Ramp	0.2%	100	400
5	Substrate feed rate	Step	-15%	40	300

Table 5. Comparison of alarm time by using different monitoring methods

Methods	Fault 1	Fault 2	Fault 3	Fault 4	Fault 5
True value	60	100	200	200	80
Single PCA	60	150	400	234	600
IRF	60	150	400	218	164
KICA-PCA	60	122	266	218	158
SSPP	60	150	400	218	164
WNSSPP ($\alpha = 5.2$)	60	132	206	206	103

Table 6. Detection delay for the five faults by using different methods in terms of SPE

Methods	Fault 1	Fault 2	Fault 3	Fault 4	Fault 5
Single PCA	0	41	188	44	472
IRF	0	35	160	12	332
KICA-PCA	0	29	111	9	212
SSPP	0	33	156	12	321
WNSSPP ($\alpha = 5.2$)	0	25	71	5	83

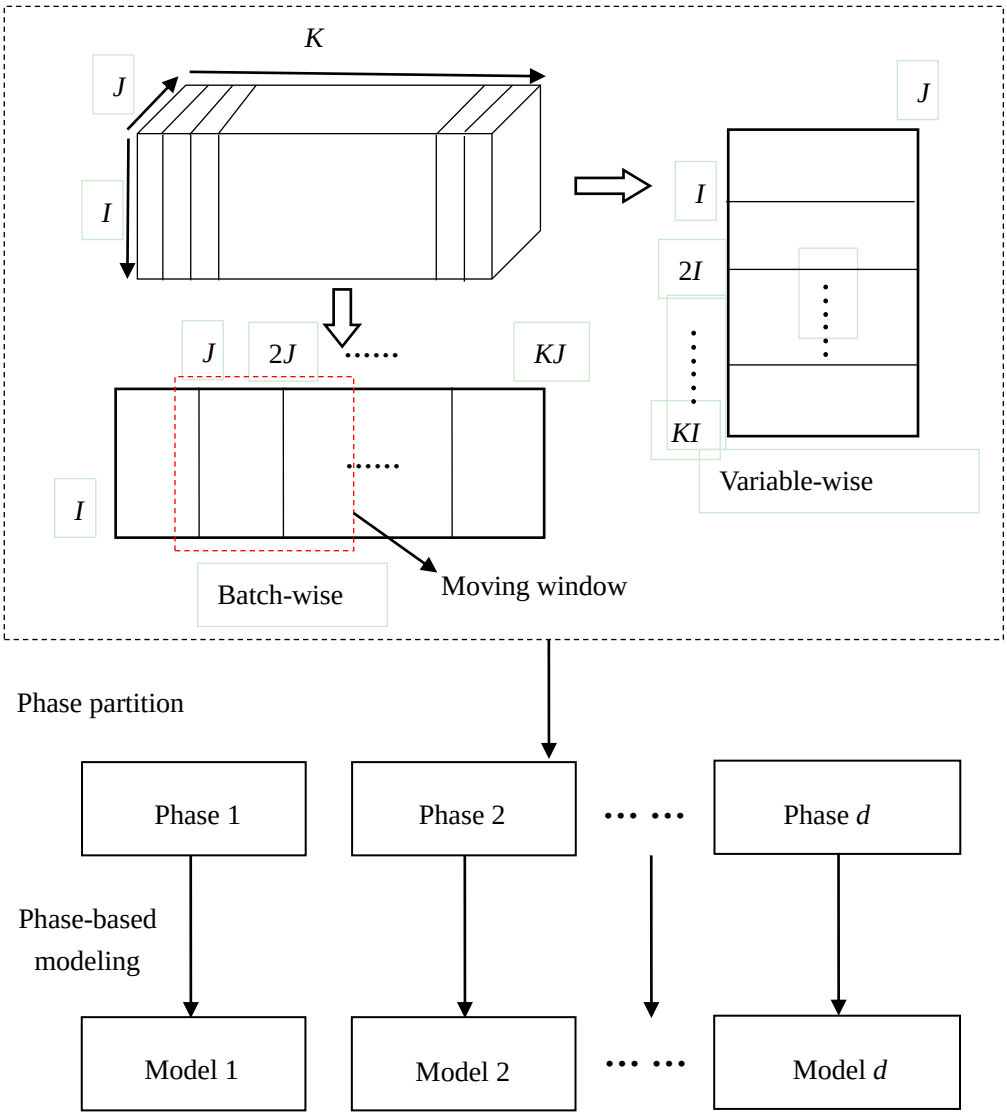

Figure 1. Illustration of multiphase batch processes monitoring

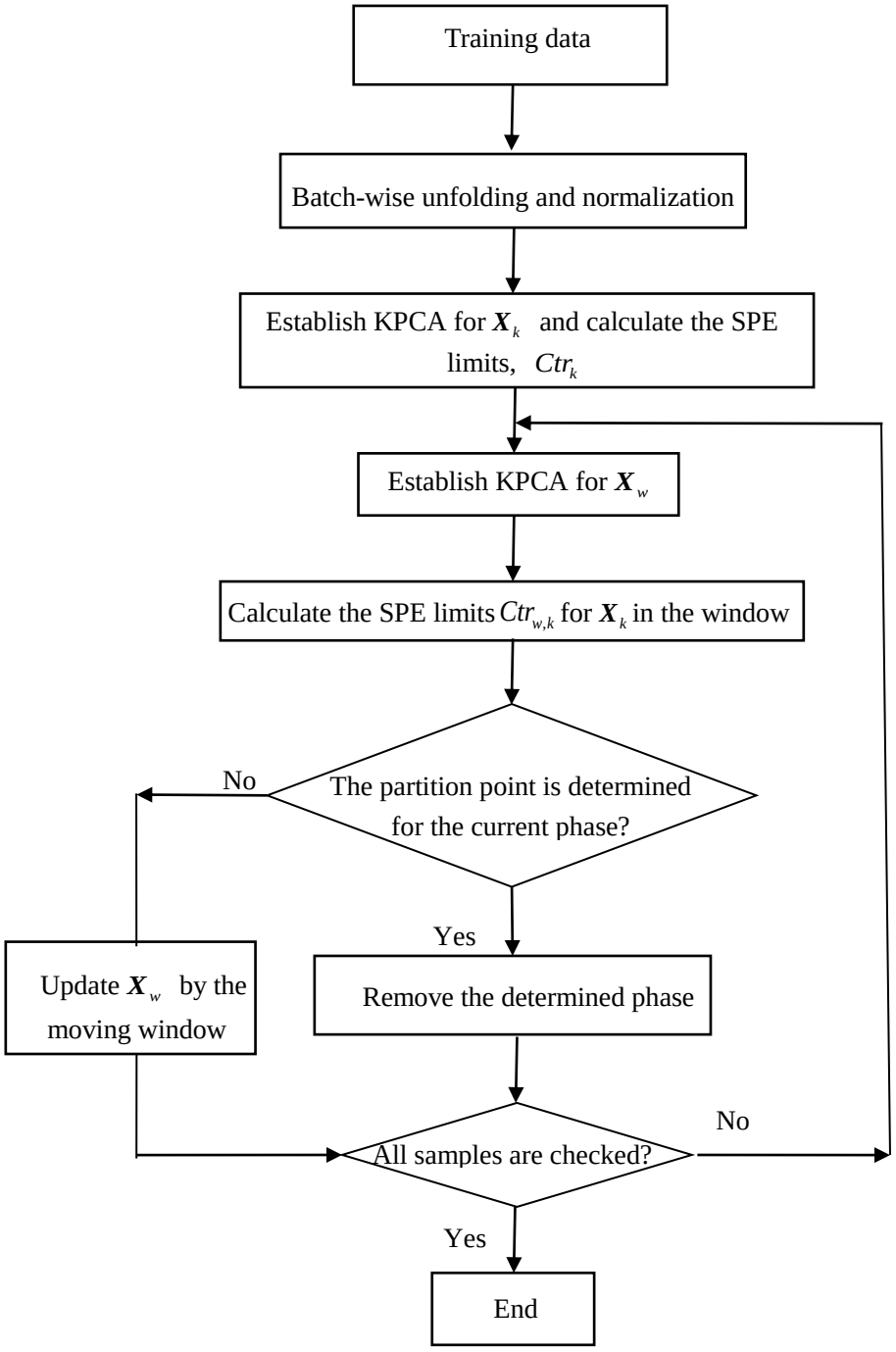

Figure 2. Flow diagram of the proposed WNSSPP-E algorithm for even-length batch processes

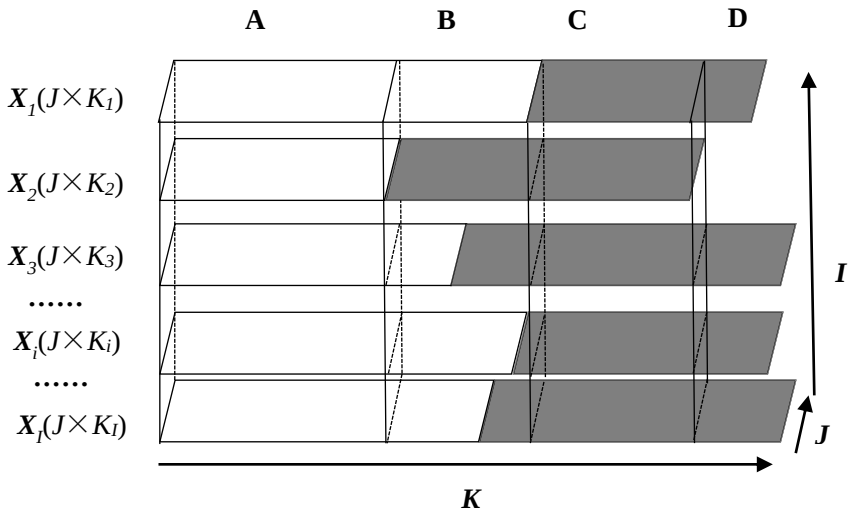

Figure 3. Graphical illustration of an uneven-length batch processes

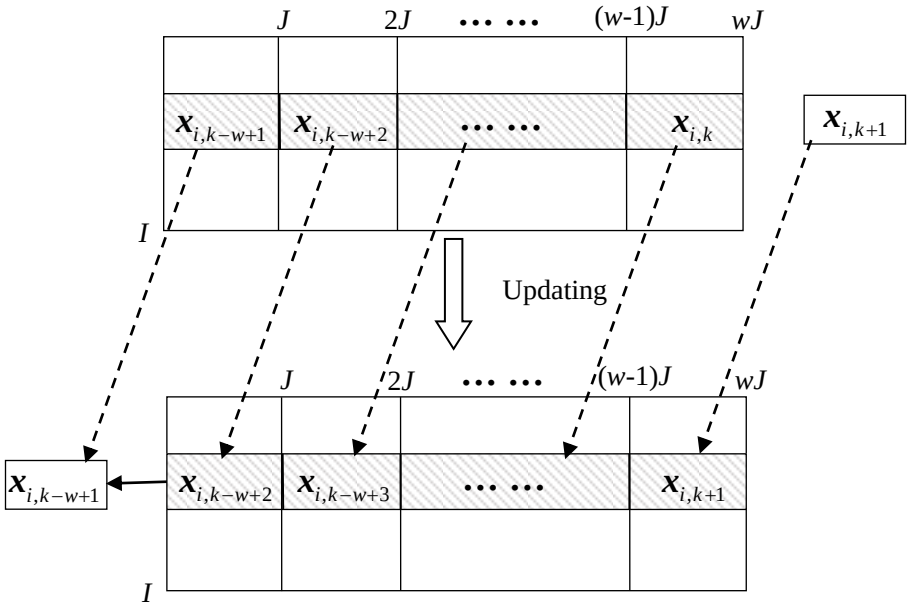

Figure 4. Graphical illustration of the updating strategy for the *i*th batch of the proposed WNSSPP-U algorithm

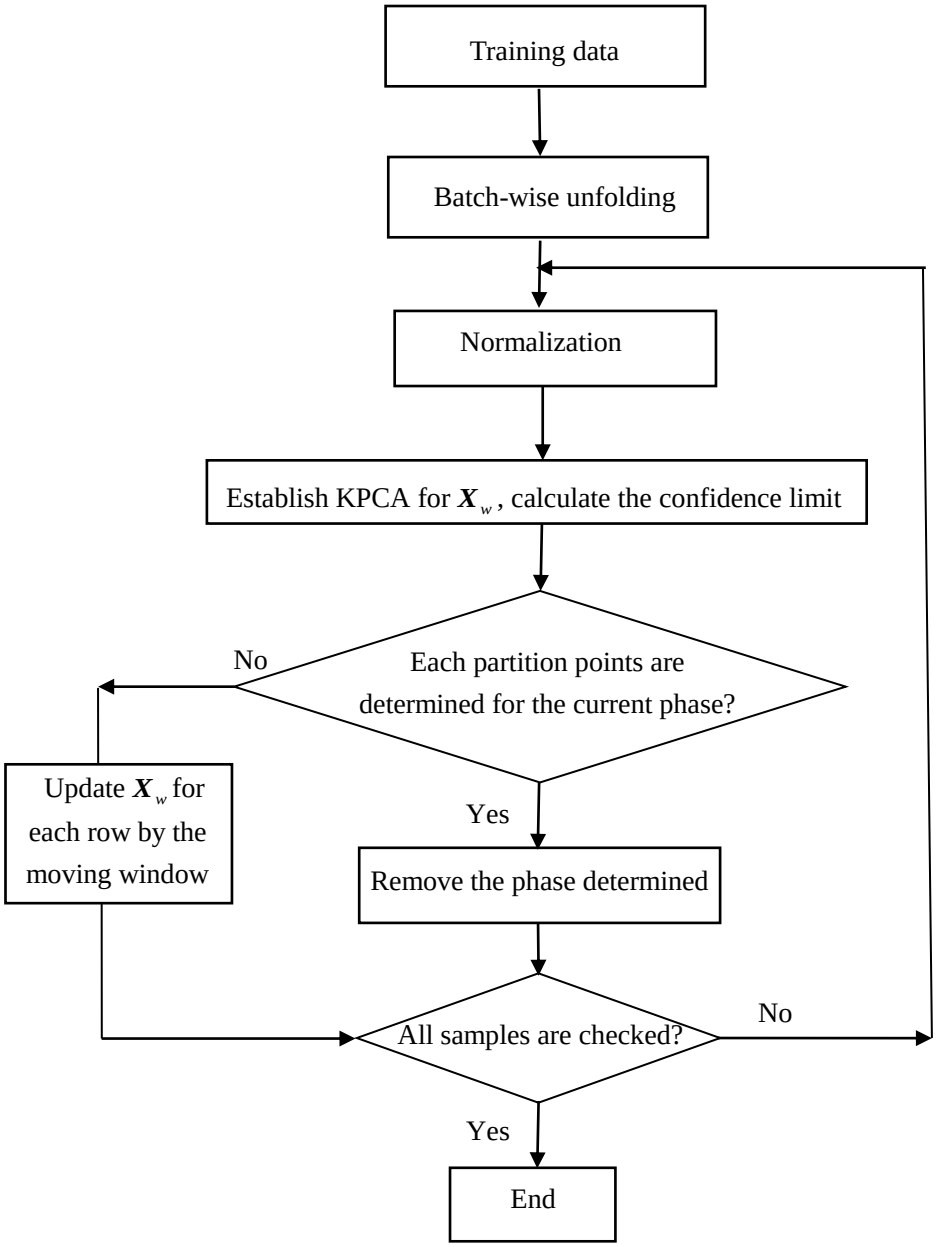

Figure 5. Flow diagram of the proposed WNSSPP-U algorithm for uneven-length batch processes

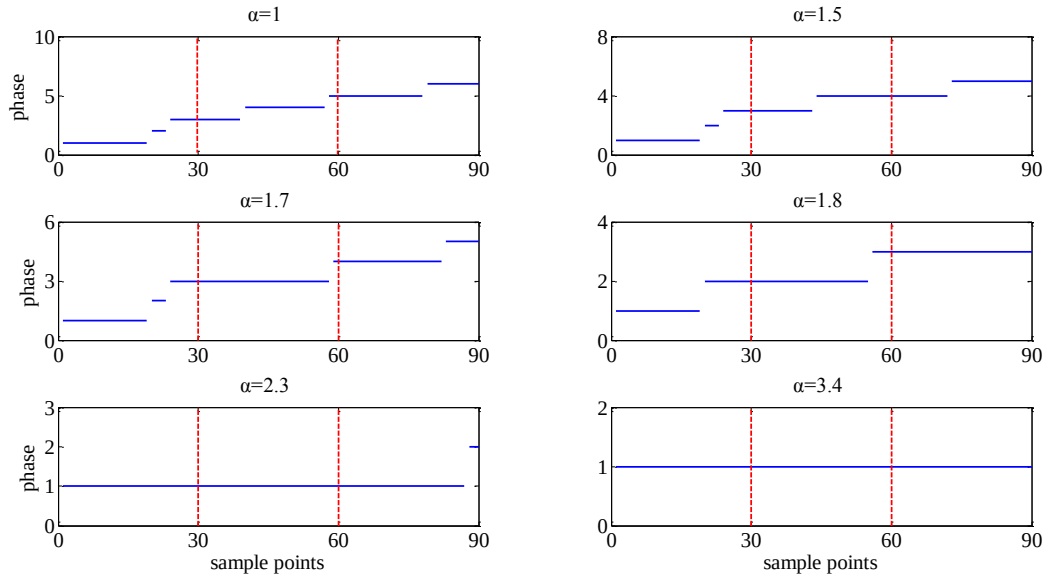

Figure 6. Phase partition results for numerical case using the SSPP algorithm

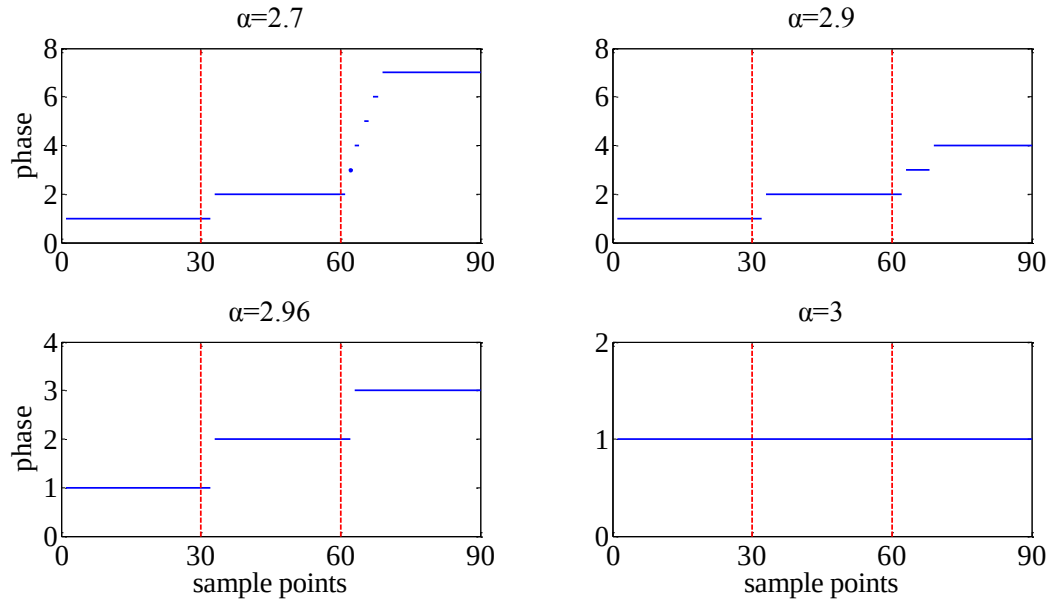

Figure 7. Phase partition results for numerical case using the proposed WNSSPP-E algorithm

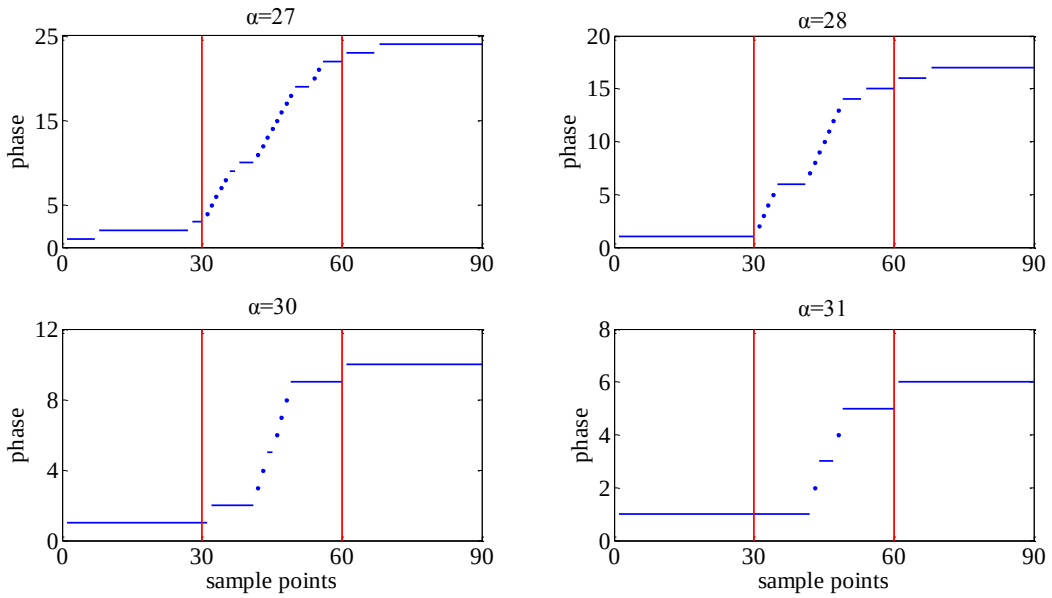

Figure 8. Phase partition results for the numerical case by using the proposed WNSSPP-E algorithm based on variable-wise unfolding

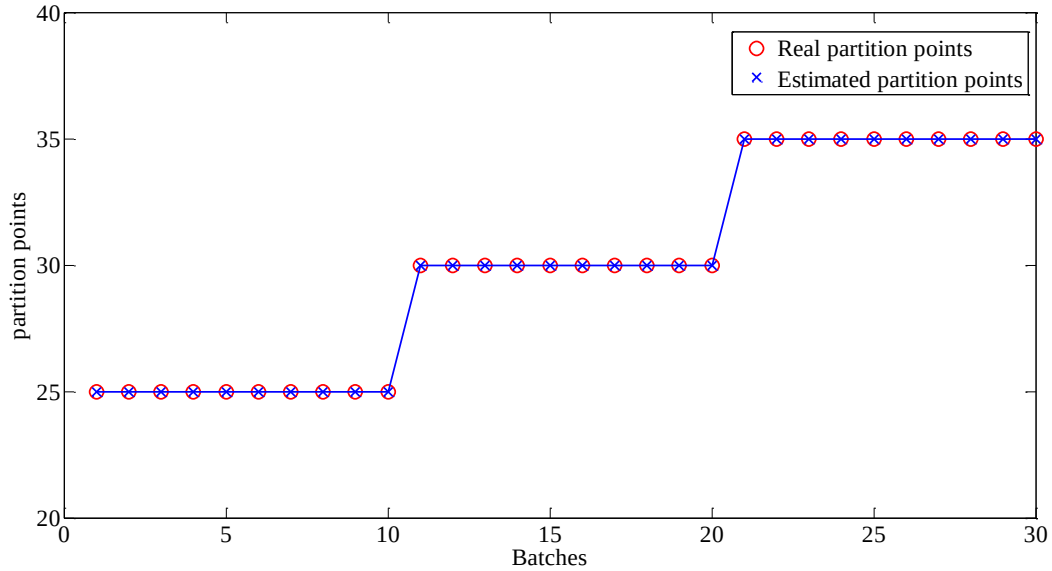

Figure 9. Partition results for the first phase of the uneven-length numerical case by using the proposed WNSSPP-U algorithm

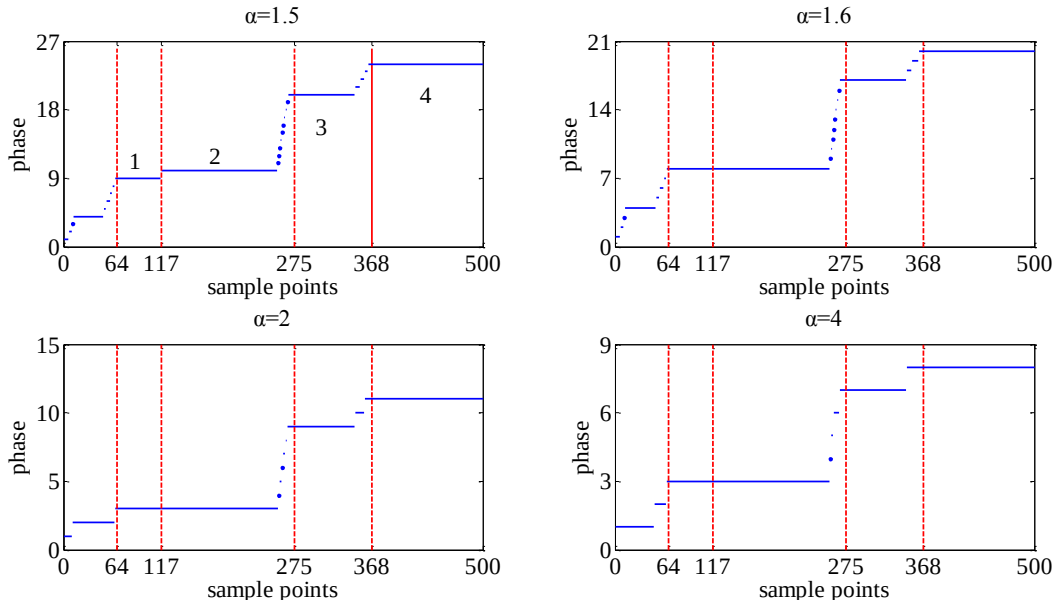

Figure 10. Phase partition results for an IM process using the SSPP algorithm

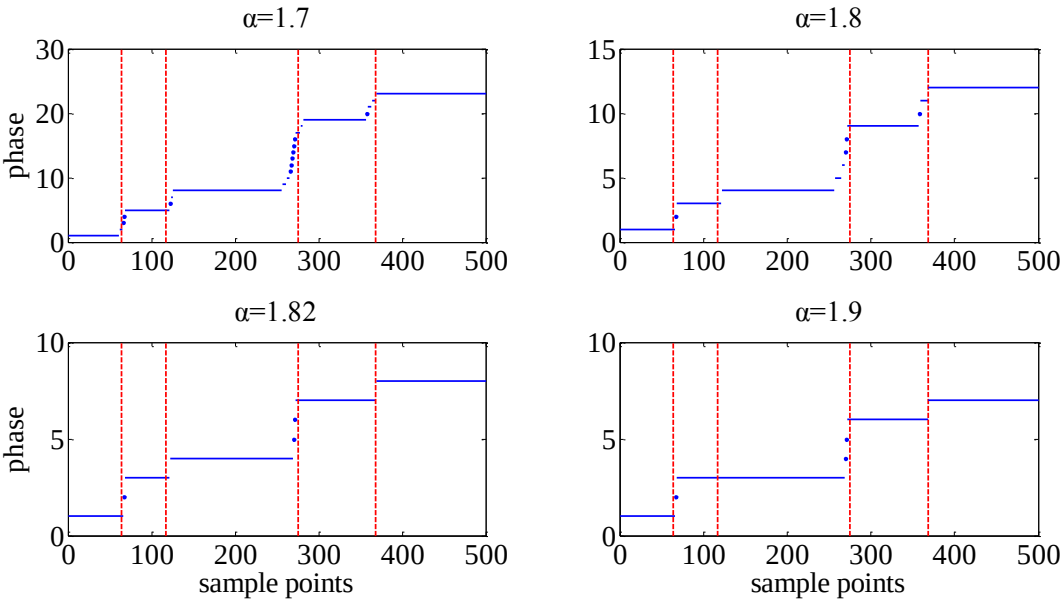

Figure 11. Phase partition results for an IM process using the proposed WNSSPP-E algorithm

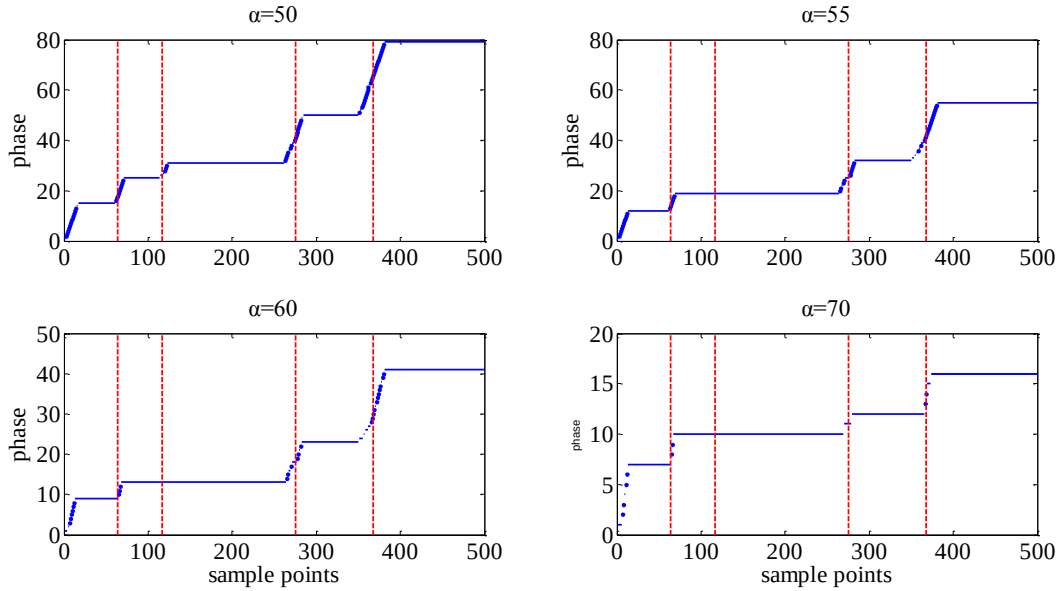

Figure 12. Phase partition results for IM process by using the proposed WNSSPP-E algorithm based on variable-wise unfolding

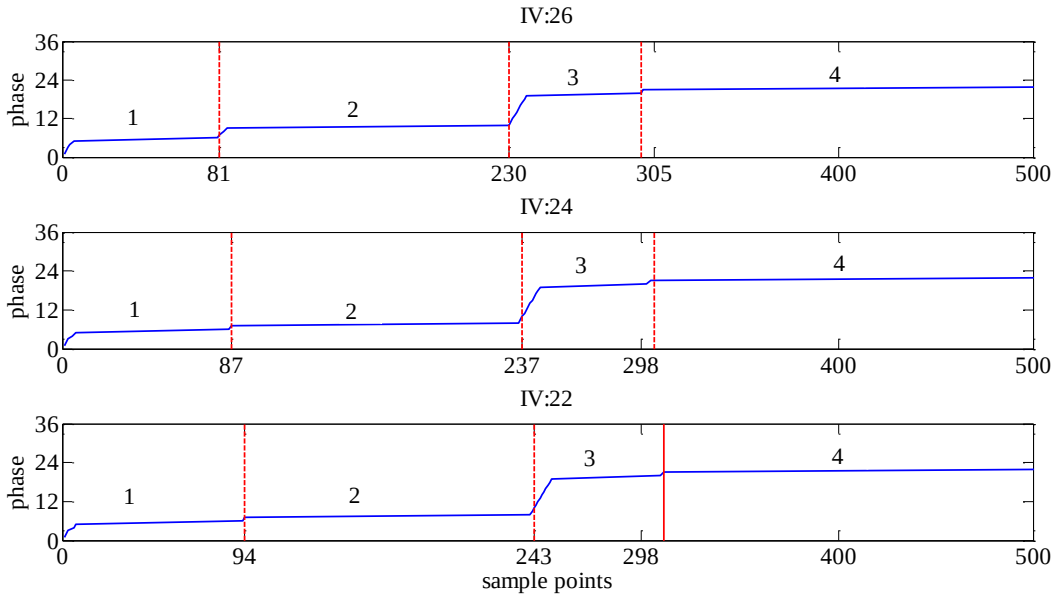

Figure 13. Phase partition results for uneven-length IM batches by using the proposed WNSSPP-U algorithm

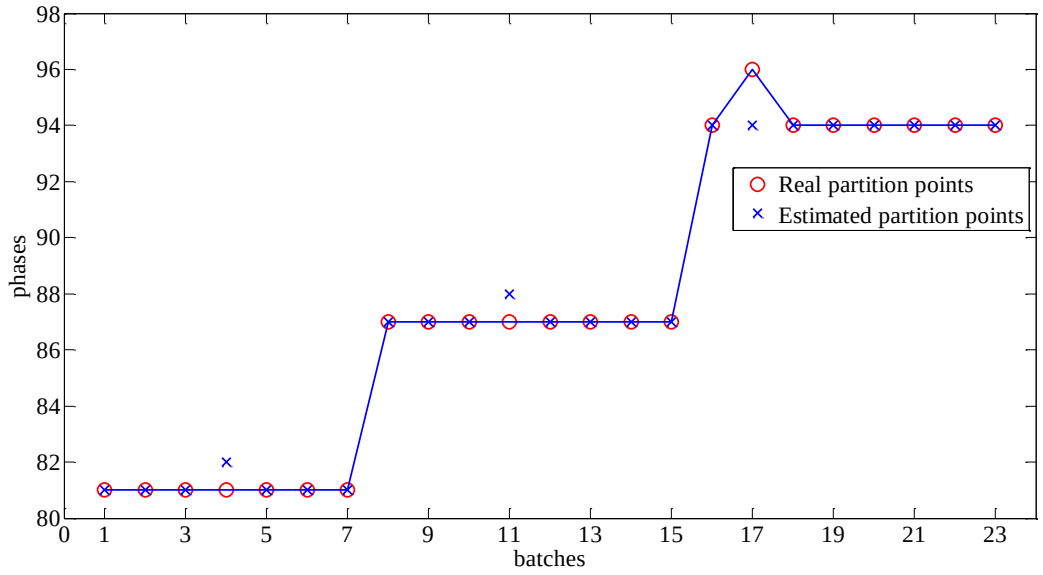

Figure 14. Partition results for the first phase of uneven-length IM batches by using WNSSPP-U algorithm

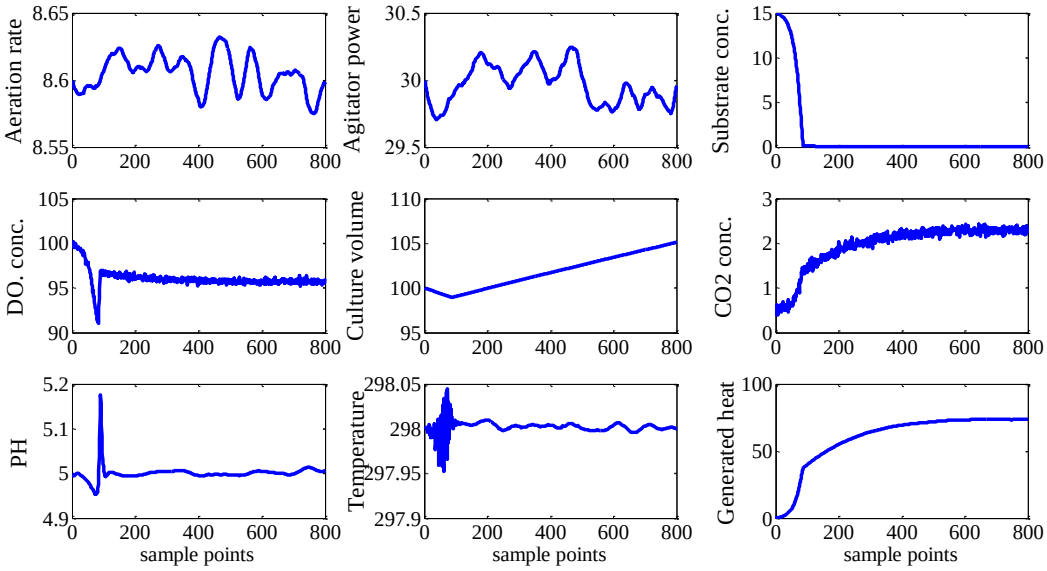

Figure 15. Trajectories of nine variables from a normal batch run of PenSim 2.0

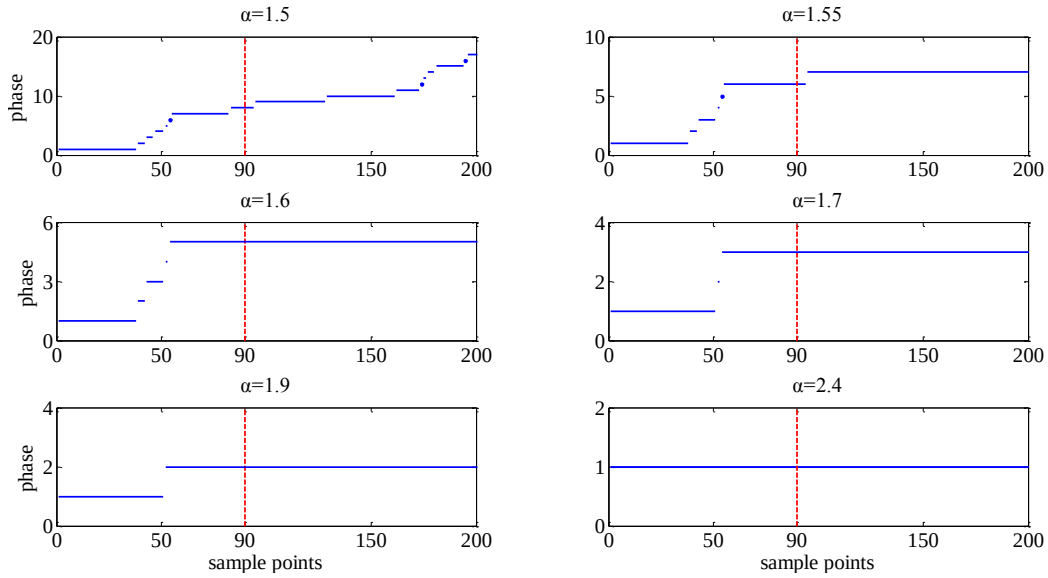

Figure 16. Phase partition results for a penicillin fermentation process using the SSPP algorithm

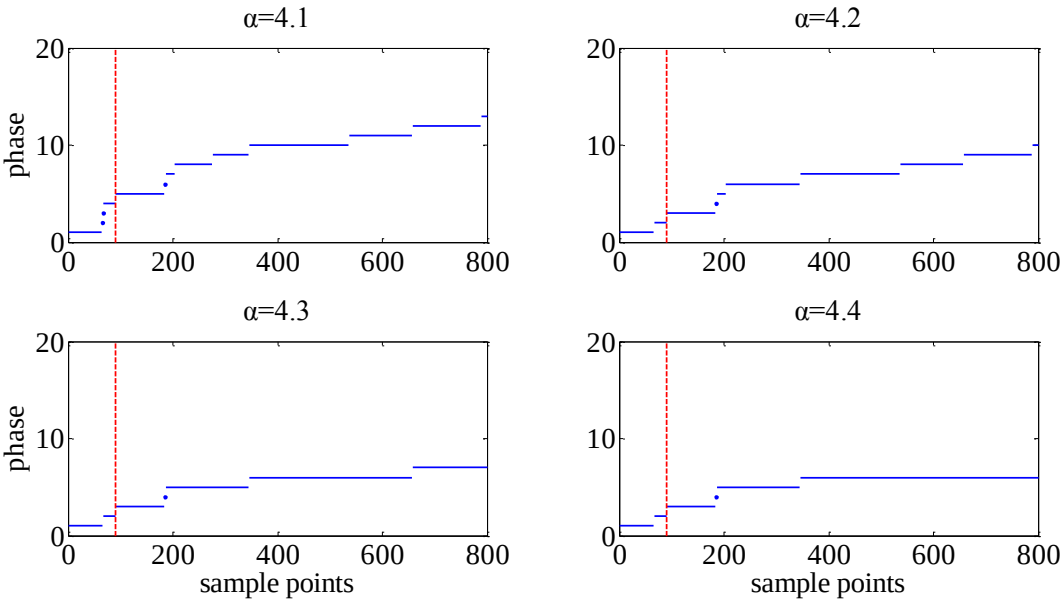

Figure 17. Phase partition results for a penicillin fermentation process by using the proposed WNSSPP-E algorithm

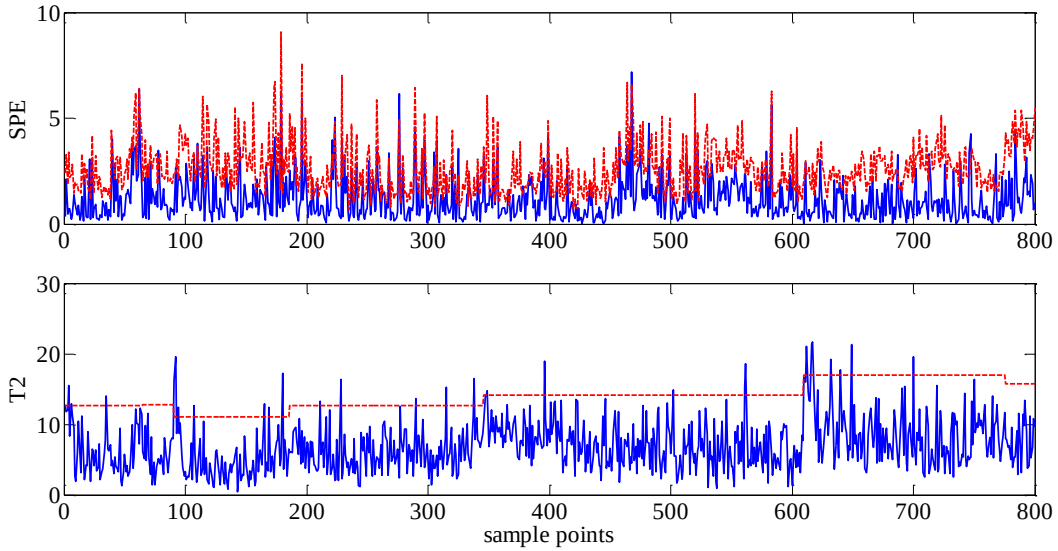

Figure 18. Online monitoring results for normal test data of a penicillin fermentation process

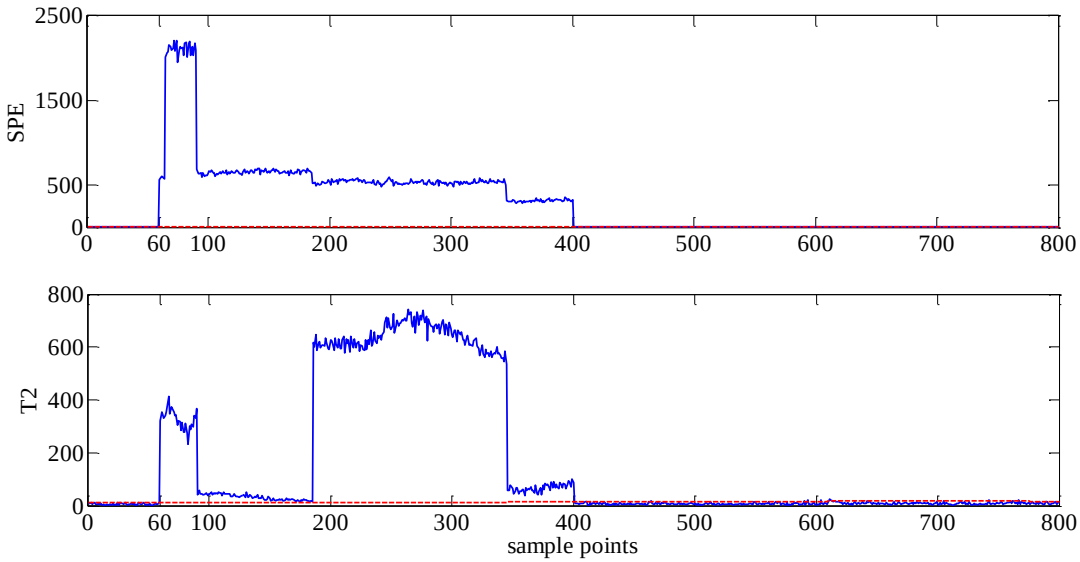

Figure 19. Online monitoring results for fault 1 of penicillin fermentation process

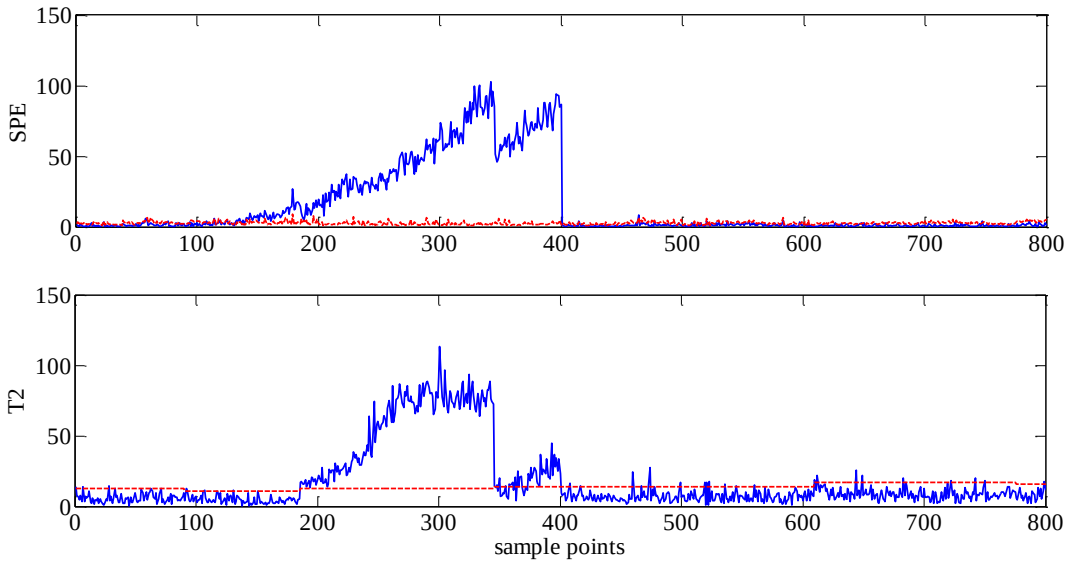

Figure 20. Online monitoring results for fault 2 of penicillin fermentation process